REMARKS/ARGUMENTS

Claims have been amended to more consistently recite "runtime," instead of "run-time." It should be noted that NO substantive claim amendment has been made. Accordingly, it is respectfully submitted that this amendment should be entered by the Examiner. The Examiner's rejection of claims under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,330,709 (*Johnson et al.)* is traversed for the following reasons:

a) ***Johnson et al.* does NOT teach or suggest: analyzing a class file for one or more runtime attributes associated with runtime performance of a Java class file** (Claim 1)

In the Final Office Action, the Examiner has asserted that *Johnson et al.* "fairly suggests" these features. To support this assertion, the Examiner has made reference to Fig. 3, and specific sections of *Johnson et al.* (Final Office Action, page 2) which are reproduced below:

> *Once the Persistent Container object has been created, an application can call methods on the static Factory class to request creation of a Java object (step 330). It should be understood that the object can be a temporary object to be stored in process local storage, or a permanent object to be stored in SAS storage. The present invention implements process local storage as an area of SAS so that all objects references will have a 128 bit address. Objects stored in process local storage will be removed once the process that created them terminates (Col. 17, lines 13-21).*
>
> *A preferred embodiment of the present invention also modifies portions of the JVM to work in conjunction with shared address space (SAS). Specifically, in the preferred embodiment the JVM can store objects in either temporary local storage or in permanent SAS storage 220. However, current JVM provide only the ability to crate transient Java objects, meaning that objects could only be stored in temporary storage. These transient objects would be deleted from memory as soon as the application which created them was removed from memory. The preferred embodiment modifies the interpretation of bytecodes associated with objects, classes, classloaders and handles to interact with temporary and persistent SAS storage since the native bytecodes were designed to interact only with temporary storage (Col. 20, lines 7-20).*

Clearly, Fig. 3 or the sections of *Johnson et al.* which have been cited by the Examiner and have been reproduced above do NOT teach or even remotely suggest:

analyzing a class file for one or more runtime attributes associated with runtime performance of a Java class file.

Moreover, it is respectfully submitted that *Johnson et al.* cannot possibly teach or suggest these features because, among other things, *Johnson et al.* pertains to a "Class Encapsulator" class and an "Object Encapsulator" class that are designed to respectively encapsulate class data and instance data (*Johnson et al.*, Abstract). This encapsulation is used to facilitate a virtual machine implementation for shared persistent objects (*Johnson et al.*, Title). It is very respectfully submitted that sharing of persistent data described by *Johnson et al.* does NOT teach or even remotely suggest analyzing a Java class file for a runtime attribute associated with runtime performance of a Java class file.

b) ***Johnson et al.* does NOT teach or suggest: generating a runtime attribute for a Java bytecode prior to loading or execution** (Claim 1)

Initially, it is respectfully submitted that the Examiner has NOT even addressed generating a runtime attribute for a Java bytecode which has been marked by analyzing of a Java class file. Instead, the Examiner has merely asserted that *Johnson et al.* teaches marking bytecodes and reading them from memory (Final Office Action, page 2). Clearly, marking and reading from memory (*Johnson et al.*, Col. 6, lines 48-56) does NOT teach or even remotely suggest the specific claimed feature of: generating a runtime attribute for each one of the Java bytecodes which have been marked based on analyzing a class file. Moreover, it is respectfully submitted that *Johnson et al.* cannot possibly teach or suggest these features because, among other things, *Johnson et al.* does NOT even relate to runtime performance associated with execution of a Java class file.

c) ***Johnson et al.* does NOT teach or suggest: reading a runtime attribute and loading into a virtual machine a runtime feature associated with the runtime attribute** (Claim 1)

It is respectfully submitted that these additional features have NOT been addressed by the Examiner. Moreover, it is respectfully submitted that in view of the grave deficiencies of *Johnson et al.* with respect to customizing a Java runtime environment based on analyzing, marking and generating a runtime attribute, *Johnson et al.* cannot possibly teach or even remotely suggest: loading a runtime feature by reading a runtime attribute which has been generated based on analyzing of a class file. In other words, *Johnson et al.* cannot possibly teach or even remotely suggest these claimed features in context of customizing a Java runtime environment based on analyzing, marking and generating of a runtime attribute.

CONCLUSION

Based on the foregoing, it is submitted that all pending claims are patentably distinct over the cited art of record. Additional limitations recited in the independent claims or the dependent claims are not further discussed because the limitations discussed above are sufficient to distinguish the claimed invention from the cited art. Accordingly, Applicant believes that all pending claims are allowable and respectfully requests a Notice of Allowance for this application from the Examiner.

Applicants hereby petition for an extension of time which may be required to maintain the pendency of this case, and any required fee for such extension or any further fee required in connection with the filing of this Amendment is to be charged to Deposit Account No. 500388 (Order No. Sun835). Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP

R. Mahboubian
Reg. No. 44,890

P.O. Box 70250
Oakland, CA 94612-0250
(650) 961-8300